A Cascading Framework for Uncovering Spammers in Social Networks

Zejia Chen, Jiahai Yang, Jessie Hui Wang
Tsinghua National Laboratory for Information Science and Technology
Dept. of Computer Science and Technology, Tsinghua University
Beijing, 100084, China
zejiachen@gmail.com, {yang,hwang}@cernet.edu.cn

Abstract—**With tremendous popularity, OSNs have become the most important platform for marketing and advertising during the past years. Meanwhile, spamming has already become a very serious problem in OSNs, drawing the attention of both academic and industry communities. In this paper, we investigate the problem of spammer detection from the perspective of user behaviors, including relation creation, user activeness, user interaction and tweet content. We quantitatively explore their correlations with spammer detection and find that tweet content is the most important factor for spammer detection, followed by relation creation. Based on these behavior factors, we propose a novel cascading framework CWB-SPAM for spammer detection in OSNs. Experiments on dataset crawled from Sina Microblog show that the proposed algorithm outperforms over all classical algorithms we investigated in terms of *F-score*[1]. Experiments also demonstrate that as a probabilistic classification model, the proposed CWB-SPAM has a good ranking quality. It enables the OSN operators to make tradeoff between precision and recall easily so that the proposed algorithm can be used in different scenarios. Besides, we also note that the proposed framework can be used in other probabilistic binary classification models and thus applied in more scenarios.**

Keywords—social network; user behavior; spammer detection; cascading framework

I. Introduction

As social networks gain tremendous popularity in the past years, spammers start to utilize OSNs as a new platform to conduct their malicious behaviors. According to a survey of Harris Interactive, 80% of OSN users have received unwanted friend request, messages and postings in their OSN accounts in the past years, which means spammers are now very popular in social networks [1]. Many users have complained about their terrible experiences in OSNs where they have to face with too much spam. Moreover, spamming in OSNs has harmful effects on both OSN users and operators. Virus, phishing or malwares contained in spam can always lead to users' treasure loss or privacy disclosure. Users may then reluctantly choose to leave and therefore the population of OSNs suffers a decrease.

Considering the great impact spammers in OSNs bring, both industry and academia have been making efforts to detect them. In industry, in order not to mistake normal users for spammers, OSN operators choose to apply simple strategies with high precision like URL blacklist and single-feature-threshold classifiers [2], although they have a much lower recall. Since these approaches exploit few aspects of behaviors, spammers can adapt their behaviors to evade detection easily. In academia, researchers notice that spammers will form tightly connected communities in OSNs [3,4]. They then propose detection approaches exploiting this social-graph characteristic. However, all these approaches rely on the assumption that the network is fast-mixing, which has been validated to be untrue in large scale OSNs like Facebook, Youtube, etc[5]. Recently, researchers start to detect spammers by training classifiers on behavior features using traditional classification algorithms like Random Forest. However, experiments in this paper reveal that these traditional algorithms don't work well for spammer detection in OSNs. In summary, although much work has already been done by academic and industry communities, spammer detection remains an unsolved problem. Actually, according to research report published by NexGate, during the first half of 2013, there has been a growth of 355% in spam in OSNs including Facebook, Twitter, Google+ and Youtube [6].

In this paper, we propose and evaluate a novel algorithm CWB-SPAM for spammer detection in OSNs. We build a classifier based on features extracted from various aspects of user behaviors so as to make the detector more solid, as it makes it much tougher for spammers to adapt their behaviors. To improve the classifier's precision and recall, we propose a cascading framework, in which we train a classifier using all data and a second classifier on instances difficult to classify. We then use these two classifiers to detect spammers in a cascading framework. Besides, we design CWB-SPAM as a probabilistic model so that OSN operators can easily make tradeoff between precision and recall in different scenarios. Experiments show that the proposed algorithm achieves a precision and recall of about 90%. In addition, by changing the parameters in CWB-SPAM, we can achieve a precision of 95% with a recall of 81% or a higher recall with a lower precision.

In summary, we frame our contributions as follows:

- After investigating various aspects of user behaviors, we find that spammers distinguish from normal users in behavior patterns of *relation creation, user interaction, tweet content* and *user activeness*. We find that they all have predictive power but cannot be used independently to detect spammers effectively.

[1] F-score refers to the harmonic mean between precision and recall.

ISBN 978-3-901882-58-6 © 2014 IFIP

- We find that among all behavior characteristics, tweet content is the most important factor for spammer detection, followed by relation creation behaviors. We also find that *tweet-related* behaviors speak louder than *relation-related* behaviors in spammer detection.
- We propose a novel probabilistic classification algorithm CWB-SPAM for detecting spammers in social networks. Experiments show that it outperforms over all other algorithms we investigated, including Bayesian, Random Forest and so on. CWB-SPAM improves the F-score from about 80% to 90%.
- As far as we know, our research work is the first one to give attention to the users' probabilities to be spammers in social networks. Experiments show that the proposed algorithm produces an accurate ranking of the users' probabilities to be spammers so that OSN operators can easily make tradeoff between precision and recall in different scenarios.

The rest of the paper is organized as follows. Section II presents an overview of previous research work. Section III investigates the behavior patterns of spammers and their correlations with spammer detection. Section IV introduces a novel algorithm CWB-SPAM incorporating these behavior features together to detect spammers. Section V conducts experiments on data from Sina Micrologand then evaluates the algorithm's performance. Section VI concludes the paper.

II. Related Work

As a long-existing problem, spammer detection has raised the attention of researchers and many efforts have been made to solve it. Most of the research work can be grouped into two categories: social-graph-based approaches and user-behavior-based approaches.

Social graph is the key element of social networks and therefore can be used to solve most problems in OSNs. Researchers have proposed many algorithms based on social graph to detect Sybil or spammers in OSNs. Yu et al. report that Sybil users form tightly connected communities in the social graph. They then propose SybilGuard [7] exploiting this property to defend Sybil attacks in OSNs. Two years later, they propose another algorithm named SybilLimit [8] reducing the number of Sybil users undetected. Danezis et al. propose another algorithm SybilInfer later in [9] to solve the same problem. However, all these algorithms rely on the assumption that the network is fast-mixing, which has been validated to be untrue in large scale networks [5]. Besides, Tran et al. propose SumUp for detecting Sybil in voting system in OSNs in [10].

Recently user behavior analysis in OSNs becomes a hot topic and many researchers start to leverage user behavior analysis to detect spammers. Lin et al. [11] report that there are three representative spamming behaviors in OSNs: aggressive advertising, duplicate reposting and aggressive following. Based on features extracted from users' behaviors, Lin et al. build detectors using machine learning algorithms including Bayesian, Random Forest, SMO and so on. Wang et al. [12] integrate behavior-based features and graph-based features together and find that Bayesian classifier has the best overall performance. Benevenuto et al. [13] create 900 honey-profiles in Twitter and observe the traffic they receive. Conducting experiments using different machine learning algorithms, Benevenuto et al. find that Random Forest outperforms over all other algorithms. Besides, Yang et al. propose a threshold-based classifier using a linear combination of *invitation frequency, outgoing requests accepted, incoming request accepted* and *clustering coefficient* in [14].

Since precision is of great importance for detecting spammers, OSN operators always need more information than a binary result. A classifier with ranking will enable OSN operators to make tradeoff between recall and precision in different scenarios. Researchers have proposed some general classification algorithms with ranking but none are specially designed for spammer detection in OSNs. Webb et al. propose a model called averaged one-dependence estimators(AODE) in [15] and Jiang et al. put forward WAODE assigning different weights to classifiers in AODE in [16] .Su et al. in [17] then propose a model called probabilistic inference trees(PIT) while Harry Zhang introduces an AUC-based algorithm for learning CITrees in [18].

In summary, although researchers have done much work on it, the problem of spammer detection in OSNs is not yet well solved. Most graph-based algorithms rely on the assumption that the network is fast-mixing [7,8,9], which has been validated to be untrue in large scale OSNs[5]. Behavior-based approaches conduct classical algorithms based on behavior features and report that Random Forest or Bayesian is the best algorithm [11,12,13]. An accurate ranking of probability to be spammers is important for spammer detection, but none probabilistic models are specially designed for this problem.

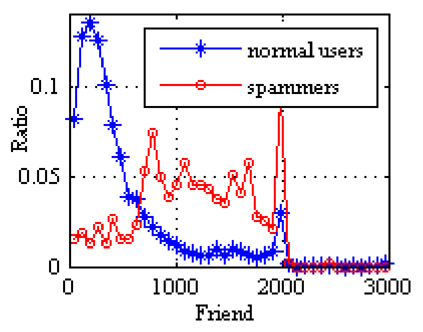
In this paper, we build a model CWB-SPAM incorporating different kinds of user behaviors. As a *behavior-based* algorithm, the assumption of fast-mixing is not needed in it. Experiments show that our algorithm outperforms over all other classical classification algorithms, including Bayesian and Random Forest. Moreover, with the behavior patterns of spammers exploited, the proposed model provides an accurate ranking of users' likelihood to be spammers.

III. Correlating User Behaviors with Spammer Detection

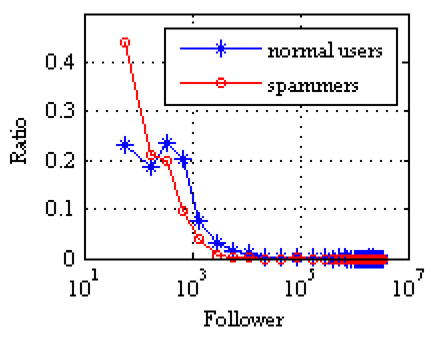
In this section, we will characterize the user behaviors in social networks and quantitatively analyze their correlations with spammer detection. Behaviors studied in this section are grouped into two categories: *relation-related* behaviors and *tweet-related* behaviors. All behaviors discussed in this section, including *friend, follower, bi-follower, friend-follower ratio, friend-bi-follower ratio* in subsection III-B, *user activeness* in subsection III-C, *repost* and *reply* in subsection III-D and *tweet length, hyperlink ratio* in subsection III-E will be leveraged as features in section IV for spammer detection in OSNs.

A. Data Used for Spammer Detection

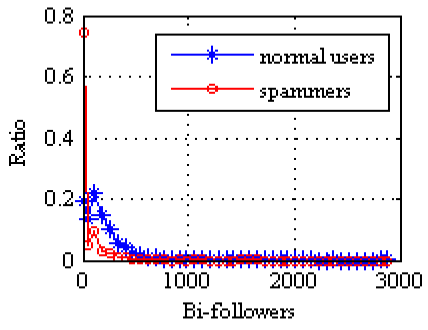
We first describe the data used for behavior analysis in this section. We randomly select a user in Sina Microblog as a start point and crawl the first two hops following it, about 9 million users in total. We then crawl their relationships so that these users form a complete sub-graph. To get more behavior



(a) Friend

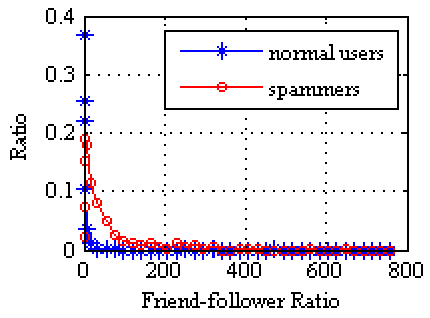


(b)Follower

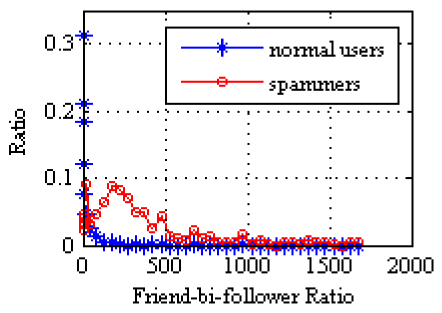


(c)Bi-follower

Fig. 1. The Distribution of Users' Followers, Friends and Bi-followers Number



(a) Friend-follower Ratio



(b) Friend-bi-follower Ratio

Fig. 2. Distribution of Friend-follower Ratio and Friend-bi-follower Ratio

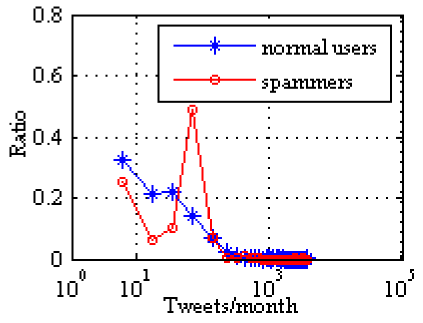


Fig. 3. Figure 3 Distribution of User Activeness

information of users in this sub-graph, we crawl the tweets they posted from March 2013 to June 2013, about 200 million in total. We sampled 4200 users from the dataset randomly and labeled them as spammers or normal users manually. Here we label those users sending aggressive following requests, malwares or advertisements in OSNs as spammers, after discussion on its definition with researchers from Sina Microblog. Among users in this dataset, 14.93% are found to be spammers. We use this dataset to conduct behavior analysis in this section and detect spammers in the next section.

B. Relation-related Behavior: Relation Creation

In this section, we will first study the correlations between spammer detection and relation creation behaviors.

We first plot the distribution of users' *friend*, *follower* and *bi-follower* in Figure 1. Let u be a user we investigate. Friend means the number of users u follows, while follower means the number of users following u, and bi-follower means the size of the intersection of u's friends and followers.

From Figure 1(a) one can find that spammers tend to follow more users than normal ones in social networks. This is mainly due to the fact that most normal users can only spend a little time in social networks and thus follow fewer people. Actually, more than 69% of them follow no more than 500 users, most of which are either famouse stars or friends in real life. In contrast, more than 88% of spammers follow more than 500 users. In fact, as spammer accounts are always created for advertising or malware spreading, spammers will always spare no efforts to reach more audience. They believe that if they follow a user, there will be probability that the user will follow back [18]. On the other hand, to behave more similarly as normal users, it is also necessary for them to create more connections with normal users. Consequently, spammers tend to follow more users in OSNs.

From Figure 1(b) we can see that compared with normal users, spammers tend to have fewer followers. However, as the distribution of *follower* of both spammers and normal users approximately follow steep power-law distribution, the difference between them is not as significant as distribution of *friend*. Fewer followers result in even fewer bi-followers, which can be seen in Figure 1(c). Unlike followers, bi-followers always indicate friendship in reality. Obviously, no one except the spammers themselves wants to make friends with spammers. When informed having a new follower, one will often check carefully before following back. Therefore, spammers are unlikely to have many bi-followers.

As noted above, spammers always tend to have more friends but fewer followers. We wonder if integrating these two features will reveal the differences better, so we plot the distribution of friend-follower ratio and friend-bi-follower ratio in Figure 2. It can be seen that spammers tend to have a bigger friend-follower ratio and friend-bi-follower ratio.

Moreover, compared with the distribution of friend-follower ratio, the difference between the spammers and normal users' friend-bi-follower ratio distribution is much more significant. There lie mainly two reasons. Firstly, we find that about 15% users of Sina Microblog are spammers in our dataset. Therefore, it's possible for spammers to follow another spammer when sending friend request randomly, which leads to a higher *friend* and lower *friend-follower ratio*. Secondly, in order to behave more likely as normal users, some spammers choose to buy 'followers' so that the friend-follower ratio becomes smaller. On the other hand, as the number of friends they can follow is limited, they always prefer not to follow back so that the *friend-bi-follower ratio* is still high. Actually, friend-bi-follower ratio can be also interpreted as the ratio of following back, which means a vote for 'normal' from others. A larger proportion of users voting for 'normal' mean a bigger probability a user to be a normal one. Therefore, for spammer detection, friend-bi-follower ratio is more informative.

C. Tweet-related Behavior: User Activeness

We define *user-activity* as the number of tweets users produce per month and present its distribution in Figure 3.We can find that spammers tend to produce more tweets than normal ones. Actually, the average user-activity of normal users is about 49 tweets/month while that of spammers is 65 tweets/month. This is due to the fact that most users will not read tweets except those in the first page. Thus, to spread advertisements or malwares, spammers will always produce tweets more frequently and therefore become more active.

D. Tweet-related Behavior: User Interaction

User interactions, including *reply* and *repost*, can also provide us information for spammer detection. Figure 4 shows the distribution of the number of replies and reposts users receive every tweet.

One can find that both the distribution of *reply* and *repost* approximately follow power-law distribution. Take *repost* for instance, near 95% spammers and 88% normal users have no more than one repost per tweet. We can also find that spammers tend to have fewer replies and reposts than normal ones. Take *reply* for example, about 50% normal users get more than one reply per tweet, while the proportion of spammers is only 3.08%.

Besides, we can find that the difference in distribution of *reply* is more significant than that in distribution of *repost*. There are mainly two reasons. Firstly, compared with reposts, replies are more likely to be interactions between close friends while spammers tend to have fewer friends. Secondly, as replies don't help to spread the advertisements or malwares, spammers always choose to repost a tweet from another spammer instead of giving a reply.

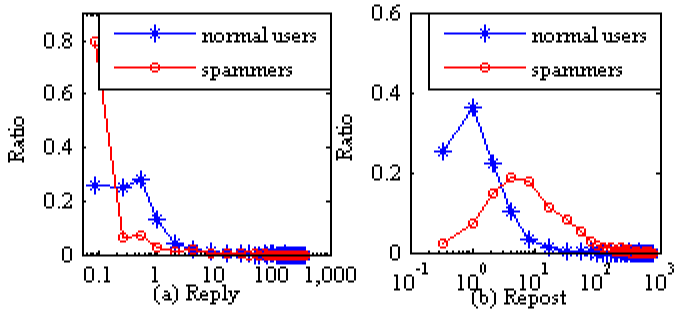


Fig. 4. Distribution of reply and repost per tweet

E. Tweet-related Behavior: Tweet Content

We further investigate the correlations between spammer detection and the tweet content features, including *tweet length* and *hyperlink ratio.* Tweet length refers to the number of bytes in a tweet while hyperlink ratio means the probability a tweet has an inline URL. Longer tweets mean more information presented while URLs always lead to shopping, games, malwares or phishing. Figure 5 shows that spammers tend to publish longer tweets so as to provide more information of what they advertise. We can also see that spammers prefer to create a tweet with a hyperlink in it. Actually, in most cases, only when users click the URLs do spammers realize their malicious goal.

Another interesting phenomenon is that if we exclude URLs in the reposted tweets, distribution of normal users and spammers' hyperlink ratio follow a similar pattern. This is mainly because as OSN operators start to detect spammers based on hyperlink ratio threshold, spammers change to repost a tweet with a URL instead of publishing a URL themselves.

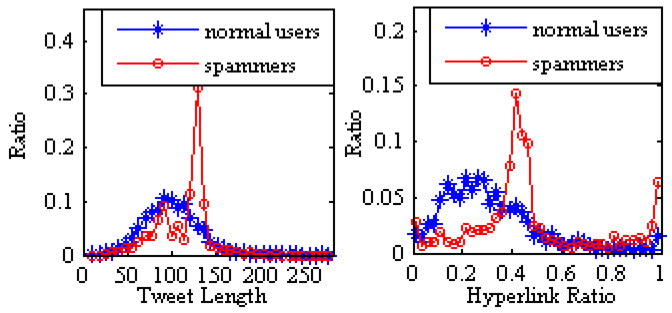


Fig. 5. Distribution of Tweet Length and Hyperlink Ratio

F. Summary

In summary, user behaviors of spammers and normal users reveal different patterns and therefore can be utilized for spammer detection. Features studied above measure different aspects of user behaviors, including relationship creation, user activeness, user interactions and tweet content. To check the predictive power of these features, we build single-feature-threshold classifiers using each feature and calculate the best F-score they can achieve, as shown in Table I. We can find that all single-feature-based threshold classifiers can only get an F-score lower than 77% and half of them can only get an F-score lower than 50%. It demonstrates that all behavior features are only weak signals and therefore cannot be used independently to detect spammers effectively.

TABLE I. F-SCORE OF THRESHOLD CLASSIFIERS

Feature	Friend	Follower	F-F Ratio	Bi-follower	F-BF Ratio
F-score	55.59%	34.67%	61.23%	62.72%	76.67%
Feature	Activity	Reply	Repost	Tweet length	Link Ratio
F-score	39.60%	43.35%	62.65%	41.12%	42.93%

IV. MODELING SPAMMER DETECTION

In the previous section, we observe that different user behaviors reveal different correlations with spammer detection. All properties have predictive power but are only weak signals. As they represent different aspects of user behaviors, we believe that integrating them together can help to detect spammers more effectively. Therefore, in this section, we will propose a novel algorithm Cascading Weighted Bayesian for Spammer detection incorporating all these properties together.

Since discretization is a necessary step in the proposed algorithm as many features used approximately follow power-law distribution, we first introduce an adaptive discretization algorithm for preprocessing these features in sub-section IV-A. We then propose Weighted Bayesian to integrate all features together in subsection IV-B and a cascading framework Cascading Weighted Bayesian to infer the users' types more effectively in subsection IV-C.

A. Preliminaries: Adaptive Discretization

As observed in Section 3, many features used for spammer detection follow approximate power-law distribution. It brings challenges for both normalization and discretization. Actually, as we show in Table II, if we partition the range in equal-width, 99.33% data will fall into the first interval of *follower* while 97.74% into first interval of *reply*. *Friend-follower ratio, friend-bi-follower ratio* and *repost* all present the same pattern. It means that these features will not work well in category classification algorithms like C4.5 and so on.

TABLE II. PROPORTION OF DATA IN FIRST INTERVAL

Feature	Follower	Reply	Friend/Follower
Ratio	99.33%	97.74%	91.29%
Feature	Friend/Bi-follower	Repost	User activeness
Ratio	83.6%	64.14%	63.84%

To solve this problem, we first normalize each feature by dividing them by their median value, so that outliers will not have a great impact. We then use an adaptive discretization algorithm to discretize these features.

In order to describe the discretization algorithm clearly, we first define some notations that will be used. Let $\mathrm{F} = \{\mathrm{f}_1, \mathrm{f}_2, \dots, \mathrm{f}_\mathrm{N}\}$ be the set of features investigated. Let $\mathrm{X} = \{\vec{x}_1, \dots, \vec{x}_\mathrm{M}\}$ be the training set and $\vec{x}_\mathrm{i} = \{\mathrm{x}_\mathrm{i}^1, \mathrm{x}_\mathrm{i}^2, \dots \mathrm{x}_\mathrm{i}^\mathrm{N}\}$ be the feature vector of user i. In addition, we set y_i as the type of user i, It can be either '*s*' or '*n*', where 's' represents spammers and '*n*' stands for normal users.

In the proposed adaptive discretization algorithm, we first partition the domain of every feature f_j into intervals adaptively. Let $\mathrm{F}_\mathrm{j} = \{\mathrm{x}_1^\mathrm{j}, \mathrm{x}_2^\mathrm{j}, \dots, \mathrm{x}_\mathrm{M}^\mathrm{j}\}$ be the set of feature f_j's value of users in the training set and $\mathrm{min}_\mathrm{j} = \mathrm{min}(F_j)$ be its minimum value and $\mathrm{max}_\mathrm{j} = \mathrm{max}(F_j)$ its maximum value. We first partition f_j's domain $(\mathrm{min}_\mathrm{j}, \mathrm{max}_j)$ into K intervals with equal width. We then divide intervals which contain data of a proportion more than α(e.g. 40%) into two intervals and update K until none intervals reach this threshold.

After running this partition algorithm, we get the intervals for discretizing feature f_j as below:

$$intervals_j = \{(from_{j1}, to_{j1}), \dots (from_{jK}, to_{jK})\} \tag{1}$$

where each interval $(\mathrm{from}_{\mathrm{jk}}, \mathrm{to}_{\mathrm{jk}})$ meets the following requirement:

$$R_{jk} = \frac{\|\{f \,/ f \in F_j,\ f \geq from_{jk},\ f < to_{jk}\}\|}{\|F_j\|} \leq \alpha \tag{2}$$

We then use these intervals to discretize the training data. Given a feature vector $\vec{x}$ of a user, we discretize the value of the j_{th} feature x^j as the midpoint of the interval it falls in.

B. Weighted Bayesian

We have observed in Section III that all features have predictive power but are only weak signals. We also find that different behaviors show different degree of correlations with spammer detection. To integrate features with different predictive power, in this section, we propose Weighted Bayesian to infer the probability a user to be a spammer.

Suppose we have discretized all data in the dataset, and let $\vec{x}$ be a discretized feature vector of a user, we estimate the probability of spammers to have a feature f_j of value x^j by:

$$p(f_j = x^j | s) = \frac{\left\|\{\vec{x}_i | \vec{x}_i \in X, x_i^j = x^j, y_i = s\}\right\|}{\|\{\vec{x}_i | \vec{x}_i \in X,\ y_i = s\}\|} \tag{3}$$

For Naive Bayesian, with naive independence assumptions, we get the probability the user to be a spammer by:

$$p(s|\vec{x}) = \frac{p(s) \prod_{j=1}^{N} p(f_j = x^j | s)}{p(\vec{x})} \tag{4}$$

As $p(s|\vec{x}) + p(n|\vec{x}) = 1$, actually $p(s|\vec{x})$ only depends on:

$$\frac{P(s|\vec{x})}{P(n|\vec{x})} = \frac{p(s)}{p(n)} * \prod_{j=1}^{N} \frac{p(f_j = x^j | s)}{p(f_j = x^j | n)} \tag{5}$$

Here we make:

$$g(s,n) = \frac{p(s)}{p(n)}, g(f_j = x^j | s, n) = \frac{p(f_j = x^j | s)}{p(f_j = x^j | n)} \tag{6}$$

Then we can get $P(n|\vec{x})$ by:

$$p(n|\vec{x}) = \frac{1}{1 + g(s,n) * \prod_{j=1}^{N} g(f_j = x^j | s, n)} \tag{7}$$

Different features help to detect spammers through the gap between the features' distribution of spammers and normal users $g(f_j = x^j | s, n)$. Actually, we can interpret the classification process from another interesting but reasonable view. Imaging the classification process as a vote for the user's type and each feature as a voter, $g(f_j = x^j | s, n) > 1$ means a vote for the user to be a spammer from feature f_j. In the contrast, $g(f_j = x^j | s, n) < 1$ means that voter f_j prefers to consider this user as a normal one. The larger gap between $g(f_j = x^j | s, n)$ and 1, the more certainty voter f_j holds and then f_j has a larger influence on the classification result.

Considering that different features help to detect spammers to various degrees, we weight different features by zooming in or zooming out the gap $g(f_j = x^j | s, n)$. If feature f_j reveals more correlation with spammer detection, it ought to play a more important role and thus its gap $g(f_j = x^j|s,n)$ is zoomed in, meaning the vote from f_j is more important. Otherwise, we zoom out its gap so that the vote from f_j becomes less important.

There comes the problem of determining the importance of the features. Considering the role gap $g(f_j = x^j|s,n)$ plays in the algorithm, we calculate the information gain of each behavior feature for spammer detection using the training data and then use the information gain vector to train model for determining the unlabeled users' types. We normalize the information gain vector W by $W = W/\mathrm{median}(W)$. Thus, $w_j > 1$ means that the correlation between f_j and spammer detection is bigger than half of the features, so we consider it more important and zoom in its gap $g(f_j = x^j|s,n)$. Otherwise, feature f_j is less important, and we zoom out its gap $g(f_j = x^j|s,n)$.

If $w_j > 1$, we zoom in the gap by:

$$g\left(f_j = x^j\middle|s,n\right)' = w_j^{\,sgn\left(g\left(f_j=x^j|s,n\right)-1\right)} * g(f_j = x^j|s,n) \quad (8)$$

Here sgn(x) refers to the signum of x. In order to keep the state of $\mathrm{sgn}(g(f_j = x^j|s,n) - 1)$, so that vote for spammers is still for spammers and vote for normal ones remains for normal ones, we zoom out the gap when $w_j \le 1$ by:

$$g\left(f_j = x^j\middle|s,n\right)' = g\left(f_j = x^j\middle|s,n\right) + (w_j - 1) * \left(g\left(f_j = x^j\middle|s,n\right) - 1\right) \quad (9)$$

According to (6), we can then get $P(n|\vec{x})$ and $P(s|\vec{x})$ by:

$$P(n|\vec{x}) = \frac{1}{1+g(s,n)*\prod_{j=1}^{n} g(f_j=x^j|s,n)'} \quad (10)$$

$$P(s|\vec{x}) = 1 - P(n|\vec{x}) \quad (11)$$

C. Cascading Weighted Bayesian for Spammer Detection

Based on *Weighted Bayesian* proposed in the previous section, we integrate different behavior features considering their correlations with spammer detection. Meanwhile, given the users' behavior feature vectors, we get their probability to be spammers, which can be also used for probability-based ranking.

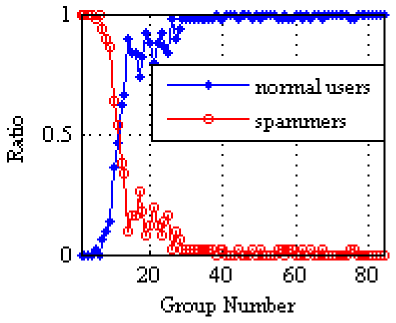


Fig. 6. User Type Distribution

To check the ranking quality of *Weighted Bayesian*, and explore patterns that might help to detect spammers, we first conduct *Weighted Bayesian* on 4200 users in the dataset and get their probability to be spammers $P(s|\vec{x})$.We rank the users by their probability to be spammers and divide the user list into 84 groups evenly. We then compute the ratio of spammers and normal users in each group, as shown in Figure 6.

Users in Figure 6 can be divided into three parts. Part one, namely the top 450 users, are whom we can confidently classify as spammers in *Weighted Bayesian,* as 99% of these users are spammers. Part two, namely the last 2500 users, are whom we can definitely classify as normal ones, as 99% of these users are normal users. Part three, the middle 1250 users, are whom we are unsure when determining their types. As the precision of classification result of part one and part two is already high enough, to improve the algorithm' s precision and recall, we just need to build a more effective model for determining the types of users in Part three, those instances difficult to classify. Here comes the novel framework CWB-SPAM for spammer detection in OSNs proposed in this paper, *Cascading Weighted Bayesian for Spammer detection.*

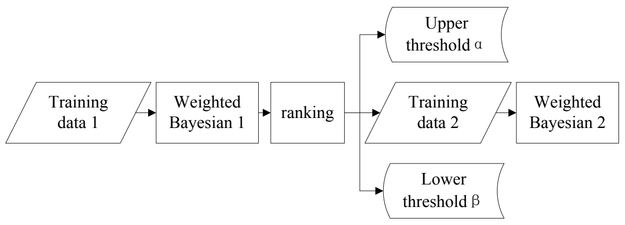


Fig. 7. Training Process of Cascading Weighted Bayesian

Figure 7 illustrates the training process of *Cascading Weighted Bayesian.* Given training data $G = \{V, X\}$, we first train a Weighted Bayesian model and thus get all users' probability to be spammers P_s. Here V refers to the set of users and X their behavior feature vectors. We then rank all users by P_s. Given confidence level c, we determine upper bound α and lower bound β so that a proportion of c of users with P_s bigger than α are spammers and a proportion of c of users with P_s smaller than β are normal ones, as shown in the (12)-(15).

$$\frac{||\{v_i|v_i \in V, P_s(v_i) \ge \alpha, y_i = s\}||}{||\{v_i|v_i \in V, P_s(v_i) \ge \alpha\}||} \ge c \quad (12)$$

$$\frac{||\{v_i|v_i \in V, P_s(v_i) > \alpha - e, y_i = s\}||}{||\{v_i|v_i \in V, P_s(v_i) > \alpha - e\}||} < c, e > 0 \quad (13)$$

$$\frac{||\{v_i|v_i \in V, P_s(v_i) < \beta, y = n\}||}{||\{v_i|v_i \in V, P_s(v_i) > \beta\}||} \ge c \quad (14)$$

$$\frac{||\{v_i|v_i \in V, P_s(v_i) \le \beta + e, y = n\}||}{||\{v_i|v_i \in V, P_s(v_i) \le \beta + e\}||} < c, e > 0 \quad (15)$$

Other users, who have P_s between α and β, are those we are not sure in the first Weighted Bayesian. We refer these data and those misclassified in the first Weighted Bayesian as $G' = \{V', X'\}$. We then train a second Weighted Bayesian based on G', as presented in Figure 7. In this paper, we set c as 0.99

so that the precision of the classification result of the first Weighted Bayesian is as high as 99%.

We then use the models trained above to predict the type of the unlabeled users, as described in Figure 8. Given an unlabeled data v_u for prediction, we first estimate its probability to be spammers P_s through the first Weighted Bayesian. If P_s is bigger than α, we consider it as a spammer. If P_s is smaller than β, we classify it as a normal one. Otherwise, we are not so sure of its type in the first Weighted Bayesian. Thus, we put it into the second Weighted Bayesian and get its probability to be spammer P_s. We then classify the user as a spammer if P_s is bigger than 0.5 and a normal one if P_s is smaller than 0.5.

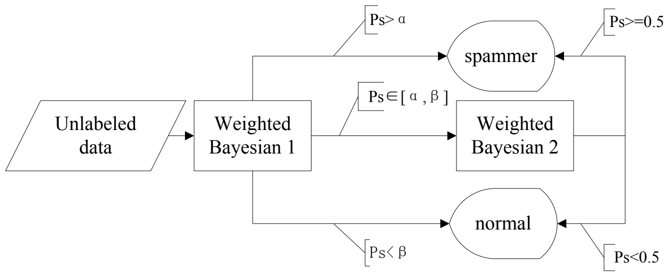


Fig. 8. Prediction Process of *Cascading Weighted Bayesian*

One should note that this framework can be also used in other probabilistic classification algorithms, including PIT, CIT, AODE and so on. There can be also another Weighted Bayesian after the second Weighted Bayesian, as the precision for spammer detection might be improved if we cascade more basic classification algorithms (e.g. Weighted Bayesian).

V. Experiment Results

Here we present the effectiveness of the proposed CWB-SPAM on spammer detection. We compare the algorithm's performance in terms of precision, recall and F-score with several classical algorithms. We then evaluate its probability-based ranking quality by computing its precision of top N users in the suspicious spammer list. Besides, we also conduct analysis on the importance of different behavior features.

A. Performance Comparison

To evaluate the effectiveness of the proposed CWB-SPAM comprehensively and fairly, we compare its performance with various baseline approaches, including representative classification algorithms of different categories and algorithms reported to work well for spammer detection.

We choose Adaboost M1 [20] as the representative of Adaboost classifier, C4.5 as the representative of decision tree, LIBSVM as the representative of SVM, Naive Bayesian and Bayesian Net [21] as representatives of Bayesian classifiers and RBF Network [22] as the representative of artificial network classifiers. We also include NBTree [23] combining Bayesian with decision tree and Random Forest [24] reported to be the best algorithm for spammer detection as baselines.

We then use 10-cross validation to evaluate these approaches and the proposed CWB-SPAM on the dataset described in Section III-A and compare their performance in terms of precision, recall and F-score in Figure 9.

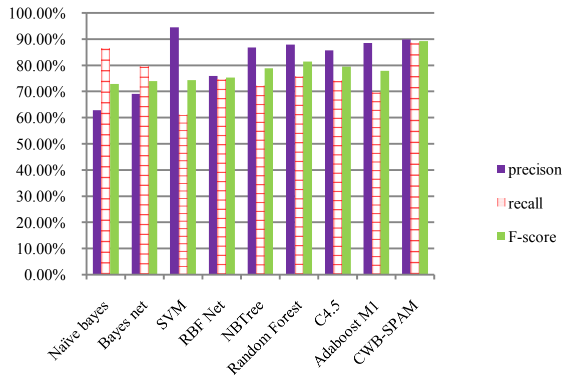


Fig. 9. Performance of CWB-SPAM & Baselines

It can be easily found that CWB-SPAM outperforms over all other algorithms in term of F-score. Actually, the F-score of CWB-SPAM is about 90% while other algorithms except Random Forest can only achieve an F-score lower than 80%. Random Forest is the best algorithm of the baselines investigated, as reported by many literatures, while CWB-SPAM improves the F-score by 10% compared with Random Forest.

We also find that the proposed CWB-SPAM achieves a better tradeoff between precision and recall. Naive Bayesian, SVM and Adaboost have a recall of 86%-94% but only a precision of 60%-70%. It means that although about 90% of spammers are detected, more than 30% 'spammers' detected are normal ones in reality. As some kind of punishment will be taken on the detected 'spammers', a precision of only 70% means that many normal users will be punished innocently. We also find that the proposed CWB-SPAM outperforms over all other algorithms in terms of both precision and recall, including Bayesian Net, RBF Network, C4.5 and NBTree.

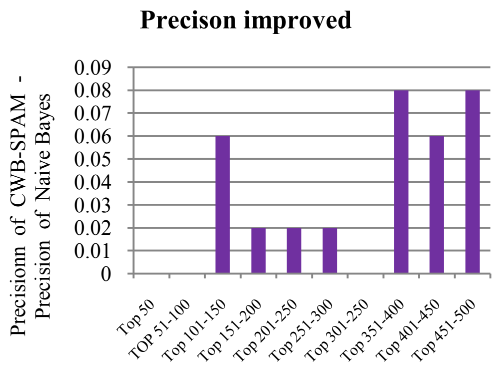


Fig. 10. Ranking Quality of CWB-SPAM vs NB

As a probabilistic classification model, CWB-SPAM can also be used for probability-based ranking. We conduct Naive Bayesian and CWB-SPAM on dataset described in Section III-A and get the users' probability to be spammers. We then get a suspicious spammer list by ranking users by their probability to be spammers. Figure 10 shows the ranking quality of the proposed CWB-SPAM against Naive Bayesian in terms of the precision of the Top N suspicious spammer list. It reveals that the proposed CWB-SPAM achieves a better ranking quality for users' probability to be spammers. Therefore, OSN operators can classify those with high rank in the list as spammers safely and make tradeoff between precision and recall easily.

B. Tradeoff between Precision and Recall

It is also valuable to check how the proposed CWB-SPAM balances between precision and recall. Through CWB-SPAM, we get the user's probability to be spammers P_s. Users are considered as spammers only when P_s is bigger than the threshold t_s. In the previous section, we set t_s as 0.5. In this section, we change t_s from 0.05 to 1 so that we can see how the proposed CWB-SPAM makes tradeoff between precision and recall, as presented in Figure 11.

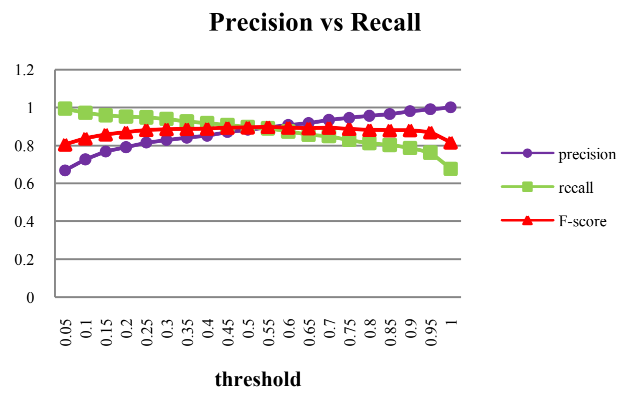


Fig. 11. Tradeoff between Precision and Recall

From Figure 11, one can find that with a higher threshold, CWB-SPAM achieves a higher precision but lower recall. We can also find that the trend of F-score remains flat while the threshold changes. F-score remains in [0.88,0.897] when the threshold varies from 0.25 to 0.8. Setting t_s 0.25 we can get a precision of 0.81 and a recall of 0.94 while a threshold of 0.8 give us a precision of 0.95 and a recall of 0.81. It means that OSN operators can easily make tradeoff between precision and recall while keeping F-score at a high level. When OSN operators want to eliminate more spammers, t_s is set lower and therefore a higher recall is obtained. When precision becomes more important for OSN operators, a higher threshold can help to improve the precision.

C. Behavior Feature Importance Analysis

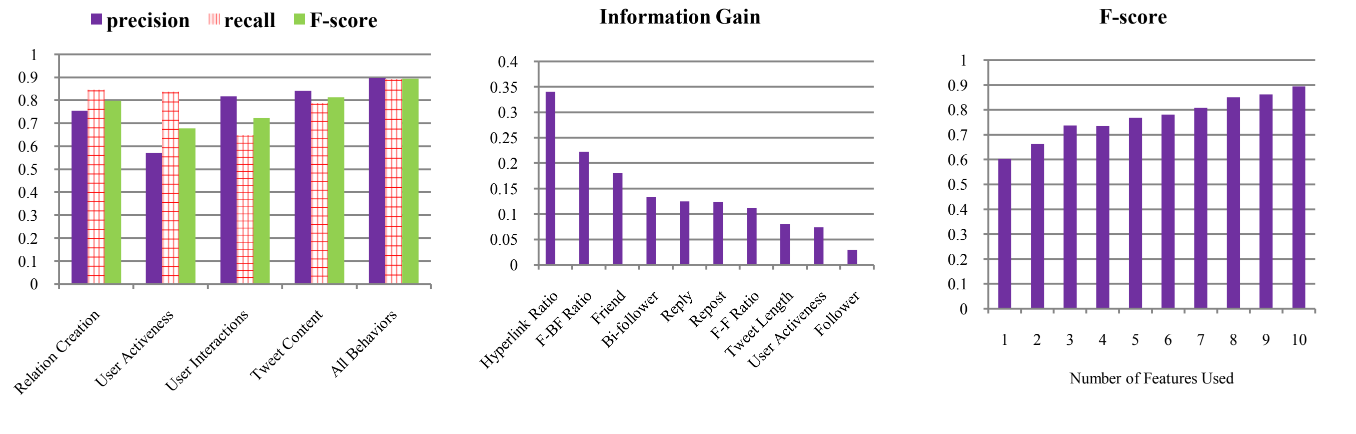
In this section, we will study the importance of different behavior features used in CWB-SPAM.

We first group the features into two categories, relation-related behaviors and tweet-related behaviors. Relation-related features include friend, follower, bi-follower, friend-follower ratio and friend-bi-follower ratio. Tweet-related features include user activeness, repost, reply, tweet length and hyperlink ratio. We first show the performance of CWB-SPAM with only relation-related behaviors and tweet-related behaviors respectively or all behaviors included, as presented in Table III. It demonstrates that tweet-related behaviors speak louder than relation-related behaviors in spammer detection in OSNs but incorporating these two categories can help to detect spammers more effectively.

TABLE III. PERFORMANCE VS FEATURS USED

Features Used	Precision	Recall	F-score
Relation-related	75.5%	84.4%	79.7%
Tweet Related	80.9%	88.7%	84.6%
All behaviors	89.7%	89.2%	89.4%

Tweet-related behaviors can be further split into three categories: user activeness, user interaction and tweet content, as we investigate in Section III-C-E respectively. Figure 12(a) shows the performance of the proposed CWB-SPAM with different categories of behaviors used. It reveals that among all behaviors, tweet content and relation creation are the most important ones. Tweet content, including tweet length and hyperlink ratio, determines how the spammers spread advertisements or malwares. Therefore, what the users speak in the tweets is always the most important factor deciding the users' types. Relation creation is the second most important factor, because only by creating more relationships can the spammers reach more audience and thus get more profit. Besides, user interactions provide more useful insights on spammer detection than user activeness, since user interactions always mean the opinions of the user' type from others.



(a) Performance with Different Behaviors Used (b) Information Gain of Different Features (c) F-score with Different Features Included

Fig. 12. Behavior Feature Importance Analysis

To further investigate the importance of the 10 features in these four categories, we compute the information gain of each behavior feature for spammer detection and present the results in Figure 12(b). It reveals that hyperlink ratio and friend-bi-follower ratio are the most informative features. Moreover, it demonstrates that friend-bi-follower ratio is more informative than friend-follower ratio, in accordance with our observations in Figure 2.

We further show the F-score by adding behavior features one by one to CWB-SPAM according to the importance of features in Figure 12(c), first add hyperlink ratio, followed by friend-bi-follower ratio, then friend, etc. We note that adding friend-bi-follower ratio and friend to the model has a larger gain on F-score than the following features. However, the difference is not so significant. In fact, adding each feature to the model will all provide a gain, on average, about 2%-3% on the F-score. This demonstrates that each behavior feature has its own contribution, as they actually measure different aspects of user behaviors.

VI. Conclusion

In this paper, we investigate the problem of spammer detection in OSNs. We first quantify the correlations between spammer detection and different behaviors, including relation creation, user activeness, user interaction and tweet content. We find that among all behaviors, hyperlink ratio is the most important factor, followed by relation creation.

Based on these behavior factors, we propose a novel spammer detection model CWB-SPAM. Experiments on dataset crawled from Sina Microblog show that the proposed model can integrate different behavior factors well and outperforms over all other classical classification algorithms for spammer detection, achieving an F-score of about 90%. One should note that the proposed cascading framework can also be used in other probabilistic binary classification models and might be applied in other scenarios. With proper features extracted, the proposed CWB-SPAM could also be used in Facebook-like OSNs. Besides, as a probabilistic classification model, CWB-SPAM enables the OSN operators to make tradeoff between precision and recall, while keeping F-score at a high level.

Acknowledgment

This work is supported by the National Basic Research Program of China under Grant No. 2012CB315806, the National Natural Science Foundation of China under Grant Nos. 61170211,61202356, 61161140454, Specialized Research Fund for the Doctoral Program of Higher Education under Grant No. 20110002110056, Tsinghua University Initiative Scientific Research Program from Ministry of Education of China, Grant No. 2012Z02151, and Joint Research Fund of MOE-China Mobile under Grant No. MCM20123041.

References

[1] G.Stringhinim, C. Kruegel, G. Vigna, Detecting Spammers on Social Networks. In: ACSAC (December 2010)

[2] Z. Chu, S. Gianvecchio, H. Wang and S. Jajodia, Who is Tweeting on Twitter: Human, Bot, or Cyborg? In: ACSAC (December 2010)

[3] S. Ghosh, G. Korlam, N. Ganguly, Spammers' networks within online social networks: a case-study on Twitter, In: WWW(March 2011)

[4] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu. Analyzing spammers' social networks for fun and profit: a case study of cyber criminal ecosystem on twitter. In: WWW (April 2012)

[5] Abedelaziz Mohaisen, Aaram Yun, Yongdae Kim, Measuring the Mixing Time of Social Graphs, In:IMC(July 2010)

[6] Nexgate, State of Social Media Spam Research Report (2013)

[7] H. Yu, M. Kaminsky, P. B. Gibbons, and A. Flaxman. SybilGuard: Defending against sybil attacks via social networks. In: SIGCOMM (August 2006)

[8] H. Yu, P. Gibbons, M. Kaminsky, and F. Xiao. SybilLimit: A near-optimal social network defense against Sybil attacks. In: IEEE Symposium on Security and Privacy (May 2008)

[9] G. Danezis and P. Mittal. SybilInfer: Detecting Sybil Nodes using Social Networks. In: NDSS (Feburary 2009)

[10] N. Tran, B. Min, J. Li, L. Subramanian. Sybil-resilient online content voting. In: NSDI (August 2009)

[11] C. Lin, Y. Zhou, K. Chen et al., Analysis and Identification of Spamming Behaviors in Sina Weibo Microblog, In: SNAKDD (August 2013)

[12] A. H. Wang, Don't follow me: Spam detection on twitter. In: SECRYPT (July 2010)

[13] F. Benevenuto, G. Magno, T. Rodgigues, and V. Almeida, Detecting spammers on twitter. In: CEAS (July 2010).

[14] Z. Yang , C. Wilson , X. Wang , T. Gao , Ben Y. Zhao , Y. Dai, Uncovering social network sybils in the wild, In: IMC (November 2011)

[15] G. I. Webb, J. Boughton, and Z. Wang. Not so naive bayes: Aggregating one-dependence estimators. Machine Learning 58, 5-24, 2005.

[16] L. Jiang , H. Zhang, Weightily averaged one-dependence estimators, In： PRICAI (August 2006)

[17] J. Su, H. Zhang, Probabilistic Inference Trees for Classification and Ranking, In: Conference of the Canadian Society for Computational Studies of Intelligence (June 2006)

[18] H. Zhang, J. Su, Learning probabilistic decision trees for AUC, Pattern Recognition Letters 29, 892–899 (2006)

[19] S. Ghosh, B. Viswanath, F. Kooti, N. K. Sharma, G. Korlam, F. Benevenuto, et al., Understanding and combating link farming in the twitter social network, In: WWW (April 2012)

[20] Y. Freund, R. Shapire, A decision-theoretic generalization of on-line learning and an application to boosting, In: Proc. of the 2nd European Conference on Computational Learning Theory, pp.23-37 (1995)

[21] G. F. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data. Machine Learning 9, 309–341(1992)

[22] D. S. Broomhead, D. Lowe, Multivariable functional interpolation and adaptive networks, Complex Systems 2: 321–355 (1988)

[23] H. Zhang, L. Jiang, J. Su, Augmenting naive Bayes for ranking, In: ICML(August 2005)

[24] T. K. Ho, The Random Subspace Method for Constructing Decision Forests. In: Transactions on Pattern Analysis and Machine Intelligence 20 (8): 832–844 (1998)